UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/657,468	09/08/2003	Makarand Gadre	MS1-1597US	9819

22801 7590 01/18/2007
LEE & HAYES PLLC
421 W RIVERSIDE AVENUE SUITE 500
SPOKANE, WA 99201

EXAMINER	
STEELMAN, MARY J	
ART UNIT	PAPER NUMBER
2191	

SHORTENED STATUTORY PERIOD OF RESPONSE	NOTIFICATION DATE	DELIVERY MODE
3 MONTHS	01/18/2007	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Notice of this Office communication was sent electronically on the above-indicated "Notification Date" and has a shortened statutory period for reply of 3 MONTHS from 01/18/2007.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

lhptoms@leehayes.com

Office Action Summary	**Application No.** 10/657,468	**Applicant(s)** GADRE ET AL.	
	Examiner Mary J. Steelman	**Art Unit** 2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 September 2003*.

2a)□ This action is **FINAL**. 2b)☒ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-28* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-28* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *08 September 2003* is/are: a)☒ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date *09/08/2003*.
4) □ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .
5) □ Notice of Informal Patent Application (PTO-152)
6) □ Other: _____.

DETAILED ACTION

1. Claims 1-28 are pending.

Specification

2. Applicant is requested to fill in the blank line of the Specification at page 1, 1st paragraph.

Information Disclosure Statement

3. IDS received 9/8/2003 has been considered.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

5. Claims 17-20 are rejected under 35 U.S.C. 101 because claims 17-20 fail to transform to a different state or otherwise produce a useful, concrete and tangible result. (In contrast, Claim 21 'generates metadata' and 'intermediate language code' and Claim 22 'translates' into machine specific binary executables, thus providing useful, concrete tangible results.)

Claim Rejections - 35 USC § 112

6. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

7. Claims 1-28 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

See MPEP 7.35.01 Trademark or Trade Name as a Limitation in the Claim

8. Claims 1, 5, 6, 8, 12, 13, 17, 19, 20, 23, and 25 contain the trademark/trade name JAVA. Where a trademark or trade name is used in a claim as a limitation to identify or describe a particular material or product, the claim does not comply with the requirements of 35 U.S.C. 112, second paragraph. See Ex parte Simpson, 218 USPQ 1020 (Bd. App. 1982). The claim scope is uncertain since the trademark or trade name cannot be used properly to identify any particular material or product. A trademark or trade name is used to identify a source of goods, namely Sun Microsystems, Inc., and not the goods themselves. Thus, a trademark or trade name does not identify or describe the goods associated with the trademark or trade name. In the present case, the trademark/trade name is used to identify/describe byte code programming language and, accordingly, the identification/description is indefinite.

Consider that Sun Microsystems, Inc. is the sole producer and/or licenser of JAVA products. The trademark JAVA identifies the source of the products and not the products themselves. In contrast, for example, C++ is a name used in trade to identify a particular nonproprietary programming language conforming to an accepted standard. Products and services incorporating the name C++ are produced by numerous sources. Further, the technologies identified using the trademark JAVA are continuously evolving. An example of this evolution can be found in "JSR 14: Add Generic Types To The Java™ Programming Language", which describes a proposed amendment to the JAVA Language Specification submitted by Sun Microsystems, Inc., in 1999 and pending approval by the JAVA COMMUNITY PROCESS Program. In view of the statements presented above, it is asserted that the trademark JAVA has no fixed definite technical meaning. Accordingly, a rejection under 35 U.S.C. 112, second paragraph, based on the use of the trademark JAVA as a limitation in a claim, is proper.

9. Claims 13 & 18 contain the trademark/trade name ".NET". Regarding claims 13 and 18, Examiner submits that the trademark ".NET" is owned by the present assignee of the instant application. As such, the present assignee is the sole producer and/or licenser of ".NET" products and has the ability to modify the technology identified under the trademark .NET . Thus, the trademark .NET identifies the source of the products and not the products themselves.

Double Patenting

10. The nonstatutory double patenting rejection is based on a judicially created doctrine grounded in public policy (a policy reflected in the statute) so as to prevent the unjustified or improper timewise extension of the "right to exclude" granted by a patent and to prevent possible harassment by multiple assignees. A nonstatutory obviousness-type double patenting rejection is appropriate where the conflicting claims are not identical, but at least one examined application claim is not patentably distinct from the reference claim(s) because the examined application claim is either anticipated by, or would have been obvious over, the reference claim(s). See, e.g., *In re Berg*, 140 F.3d 1428, 46 USPQ2d 1226 (Fed. Cir. 1998); *In re Goodman*, 11 F.3d 1046, 29 USPQ2d 2010 (Fed. Cir. 1993); *In re Longi*, 759 F.2d 887, 225 USPQ 645 (Fed. Cir. 1985); *In re Van Ornum*, 686 F.2d 937, 214 USPQ 761 (CCPA 1982); *In re Vogel*, 422 F.2d 438, 164 USPQ 619 (CCPA 1970); and *In re Thorington*, 418 F.2d 528, 163 USPQ 644 (CCPA 1969).

A timely filed terminal disclaimer in compliance with 37 CFR 1.321(c) or 1.321(d) may be used to overcome an actual or provisional rejection based on a nonstatutory double patenting ground provided the conflicting application or patent either is shown to be commonly owned with this application, or claims an invention made as a result of activities undertaken within the scope of a joint research agreement.

Effective January 1, 1994, a registered attorney or agent of record may sign a terminal disclaimer. A terminal disclaimer signed by the assignee must fully comply with 37 CFR 3.73(b).

11. Claims 1, 12, 17, and 23 are provisionally rejected on the ground of nonstatutory obviousness-type double patenting as being unpatentable over claims 1, 13, and 25 of copending Application No. 10/657463. Although the conflicting claims are not identical, they are not patentably distinct from each other because claims recite generating common intermediate language, receiving a portion of JAVA language source code referencing a first class having a definition that is uniformly applicable to a plurality of classes (generic class), the source code identifying one of the plurality of associated classes (compiling an instance of the generic class) and generating language neutral intermediate language (common intermediate language code). Although the claims are not using the identical terminology, they are addressing similar elements.

This is a provisional obviousness-type double patenting rejection because the conflicting claims have not in fact been patented.

Claim Rejections - 35 USC § 103

12. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

13. Claims 1-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 6,760,905 B1 to Hostetter et al., in view of US Patent 6,018,628 to Stoutamire.

Per claim 1:

A method of generating common intermediate language code comprising:

-writing first JAVA.TM. language source code that comprises a definition of a generic class usable in a framework;

-generating an instance of the generic class;

-compiling the instance of the generic class into common intermediate language code executable by a runtime engine.

Hostetter disclosed (col. 2:62-67) JAVA programming language with the usage of parameters within the source code definition of a class template, compilation, and (col. 3:4-15) creating a class from the class template (generating an instance of the generic class). Col. 5:58-col. 6: 9, Source code defining the class template is compiled into an object representation suitable for use as a resource when subsequently compiling classes based on the template…These source code representations may be some type of intermediate code (i.e. code categorized between source code and object code)...In general, a source code representation is an abstract representation (common intermediate language code) of the source code of the class template not yet specialized for any particular template generated class.

Stoutamire more (Abstract),explicitly disclosed generating code using parameterized classes, to generate unparatermized class code (an instance of the generic class), (col. 12: 27) generating byte code from parameterized class files. Col. 14: 64, by adding certain annotations of flags to the byte code, modified virtual machines can take further advantage . Col. 15: 30, certain embodiments have been described using the JAVA programming language, the present invention can be practices on a variety of programming languages...

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Hostetter, using the teachings of Stoutamire, to produce an invention that efficiently allocates memory (Hostetter, col. 2:65-col. 3: 3, Stoutamire, col. 2:48-64) by using parameterized classes and generating only necessary class instances.

Per claim 2:

-storing the source code in a class library of the framework.

Col. 5: 63, "resource of source code for compiling' Col. 6: 19-20, "source code declaring a class based on the class template is compiled"

Per claim 3:

-receiving second source code referencing the generic class.

Col. 6: 37-39, "source code instruction (i.e. method call instruction) that refers to a method of a template-generated class causes the generation of a method binding"

Per claim 4:

-receiving second source code referencing the generic class;

Col. 6: 37-39, "source code instruction (i.e. method call instruction) that refers to a method of a template-generated class causes the generation of a method binding" Col. 4: 27, "The runtime compiler is invoked (second source code) …

-parsing the second source code into a parse tree representing the second source code.

Col. 3: 66 – col. 4: 7, "The process for generating method bindings is triggered whenever a method call instruction, referencing a class method of a template-generated class, requires compilation. It involves (1) scanning the class representation…(2) creating a method binding; (3) compiling (create parse tree) the source code representation." Col. 4: 27, "The runtime compiler is invoked (second source code) …

Per claim 5:

-parsing the portion of JAVA.TM. source code into a parse tree representing the source code. See rejection of limitations addressed in claim 4 above.

Per claim 6:

-writing first JAVA.TM. language source code comprises defining at least one parameter associated with the generic class.

Col. 12:50, class object is defined by its parameters

Per claim 7:

-the at least one parameter is an unconstrained type.

Col. 12: 50-58

Per claim 8:

-declaring an instance of the generic class in second JAVA.TM. source code.

Col. 6:50-53, the source code representation that constitutes the body of a method is compiled when the program executes a compiled method call instruction invoking that method. The first time stub code is referenced which initiates compilation of a generic class.

Per claim 9:

-declaring an instance of the generic class comprises specifying a type from a plurality of allowable types associated with the generic class.

Col. 8: 34-41, template generated classes used in program code are data types, declared by a variable declaration statement. Parameters of the class template are replaced with actual data types.

Per claim 10:

-the specified type is another generic class.

Col. 8: 34-41 & col. 7: 23, 'inherited classes'

Per claim 11:

-the generic class comprises one of: a Queue class; a Dictionary class; and a Stack class.

As an example, col. 9:1-5, ClassObject stores binding objects as elements of a linked list (queue class).

Per claim 12:

A method of using a generic class comprising:

-adapting existing JAVA.TM. source code to include a declaration of a first generic class provided by a software package having a class definition of the first generic class;

-compiling the adapted JAVA.TM. source code with the class definition to generate common intermediate language code.

See rejection of claim 1 above. Col. 8: 24, generating object representations of template generated classes. Col. 8: 42, Compiler encounters declared type as template generated class, and class is compiled (compiling source code with class definition) into a template generated class representation.

Stoutamire more (Abstract),explicitly disclosed generating code using parameterized classes, to generate unparatermized class code (an instance of the generic class), (col. 12: 27) generating byte code from parameterized class files. Col. 14: 64, by adding certain annotations of flags to the byte code, modified virtual machines can take further advantage . Col. 15: 30, certain embodiments have been described using the JAVA programming language, the present invention can be practices on a variety of programming languages…

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Hostetter, using the teachings of Stoutamire, to produce an invention that efficiently allocates memory (Hostetter, col. 2:65-col. 3: 3, Stoutamire, col. 2:48-64) by using parameterized classes and generating only necessary class instances.

Per claim 13:

-the adapting comprises: editing the existing JAVA.TM. source code with a Visual J# .NET.TM. application in a .NET.TM. Framework.

Hostetter disclosed editing source code with an application framework (col. 3: 4 – col. 4:32)

Per claim 14:

-the class definition defines at least one parameter of the generic class.

Col. 8: 34-42.

Per claim 15:

-compiling comprises: validating a specified type of the generic class according to the class definition.

Col. 6: 29-30, col. 9: 21-30, type object defines the field in terms of what values it can store, col. 13:1-15.

Per claim 16:

-adapting comprises nesting a second generic class in the declaration of the first generic class.

Col. 11: 34, class templates are inherited (nesting)

Per claim 17:

A system for authoring source code comprising:

-a class library having a generic class definition;

-a means for receiving a declaration of an instance of the generic class in JAVA.TM. language source code.

Col. 7: 32-35, field descriptors and method descriptors stored in ClassTemplate, Col. 8: 25, ClassTemplate provides source code representation (class library having a generic class definition) for all the methods and fields defined in the class template. Col. 8: 42-46, Compiler encounters variable declaration in which the declared type is a template generated class, the class is compiled into a template generated class representation, an object representation of the template generated class. Col. 9: 65, a method binding is created for each class method that is referenced by a source code instruction. Col. 10: 39-42, compiler compiles to a type signature object (instance of the generic class)

Stoutamire more (Abstract),explicitly disclosed generating code using parameterized classes, to generate unparatermized class code (an instance of the generic class), (col. 12: 27) generating byte code from parameterized class files. Col. 14: 64, by adding certain annotations of flags to the byte code, modified virtual machines can take further advantage . Col. 15: 30, certain embodiments have been described using the JAVA programming language, the present invention can be practices on a variety of programming languages...

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Hostetter, using the teachings of Stoutamire, to produce an invention that efficiently allocates memory (Hostetter, col. 2:65-col. 3: 3, Stoutamire, col. 2:48-64) by using parameterized classes and generating only necessary class instances.

Per claim 18:

-the means for receiving comprises a computer-readable medium having stored thereon a VISUAL J# .NET.TM. application.

Col. 13: 56, 66-col. 14:1, computer readable medium storing byte code programming environment compiles source files.

Per claim 19:

-a common intermediate language importer operable to associate the generic class declaration in the JAVA.TM. language source code to the generic class definition.

Col. 10: 15-39, associates generic class declarations.

Per claim 20:

-a semantic analyzer operable to validate the generic class declaration in the JAVA.TM. language source code according to the generic class definition.

Col. 6:30-comiler bindings, col. 10: 32-36, check if method descriptors correspond to the referenced class method.

Per claim 21:

-a code generator operable to generate metadata descriptive of the generic class and further operable to generate common intermediate language code representative of the generic class.

Col. 6: 29-30, first compilation results in the type signature of a method being compiled into a method binding. Col. 6: 39-40, A method binding (meta data descriptive of the generic class) is an object that stores information about a class method. Col. 6: 24, class template representation (intermediate language)

Per claim 22:

-a runtime engine operable to translate the common intermediate language into machine-specific binary executable by a computer associated with the runtime engine.

Col. 6: 33, resulting executable code

Per claim 23:

A computer-readable medium having stored thereon microprocessor-executable instructions for performing a method comprising:

-receiving input representing a generic class definition in a JAVA.TM. language;

-receiving source code that references the generic class;

-compiling the source code with an instance of the generic class into common intermediate language code executable by a runtime engine.

See rejection of limitations addressed in claim 1 above.

Stoutamire more (Abstract),explicitly disclosed generating code using parameterized classes, to generate unparatermized class code (an instance of the generic class), (col. 12: 27) generating byte code from parameterized class files. Col. 14: 64, by adding certain annotations of flags to the byte code, modified virtual machines can take further advantage . Col. 15: 30, certain embodiments have been described using the JAVA programming language, the present invention can be practices on a variety of programming languages...

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention to modify Hostetter, using the teachings of Stoutamire, to produce an invention that efficiently allocates memory (Hostetter, col. 2:65-col. 3: 3, Stoutamire, col. 2:48-64) by using parameterized classes and generating only necessary class instances.

Per claim 24:

-storing the generic class definition in a framework class library.

Col. 5: 61, class template representation

Per claim 25:

-the source code comprises JAVA.TM. language source code.

Col. 2: 63, col. 13: 65 source files written in a language that creates the output

Per claim 26:

-the method further comprises generating metadata describing the generic class.

See rejection of limitations addressed in claim 21 above.

Per claim 27:

-the generic class definition comprises a generic class name and two angular brackets around one or more parametric types.

Hostetter failed to explicitly disclose a generic class name and two angular brackets.

However, Hostetter disclosed, (col. 6: 5) a source code representation is an abstract representation of the source code of the class template not yet specialized…the source code representations stored in these method descriptors represent a type signature and a method body…the type signature represents the arguments that the method expects and the value that the method returns. The method body is the source code constituting the functionality. A method table is generated with entries for each defined method.

It would have been obvious, to one of ordinary skill in the art, at the time of the invention, to realize that source code representations store the same information.

Per claim 28:

-the method further comprises compiling the generic class definition into common intermediate language code.

Col. 6:1-61, Abstract, intermediate code is produced. At runtime, when a class method is invoked for the first time, stub code for that method initiates further compilation of a non-generic resulting executable object.

Conclusion

14. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Wei Zhen can be reached at (571) 272-3708. The fax phone number for the organization where this application or proceeding is assigned: 571-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman

01/09/2007

Mary Steelman
Primary Examiner